AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) In a computer system that supports a serialization engine capable of generating source code for objects, a method of serializing one or more objects from an initial representation to any of one or more subsequent representations, for one or more standard object types and serialization formats, and which may be extended to cover one or more custom object types and serialization formats, the method comprising acts of:

providing a serialization manager to (i) coordinate one or more standard serialization providers that each identify one or more standard serializers for a standard object type or serialization format, and (ii) load, as needed, one or more custom serialization providers that each identify one or more custom serializers for one or more custom object types or serialization formats that are not covered by the one or more standard serialization providers;

requesting a serializer from the serialization manager for an object graph that comprises an object of a particular object type, and for a particular serialization format, wherein the serializer is requested from the serialization manager as part of a cut, copy or paste operation and such that due to the cut, copy or paste operation, the serializer produces a snippet of code sufficient to undo or redo a change to the object graph, but without producing a class representation of the object graph; and

calling the serializer to serialize the object graph.

2. (Original) A method as recited in claim 1, wherein the serializer is a custom serializer.

3. (Original) A method as recited in claim 2, wherein the serialization manager automatically loads the one or more custom serialization providers in response to the request.

4. (Original) A method as recited in claim 1, wherein the computer system also supports a visual user interface designer and the serialization engine is capable of generating source code for one or more user interface objects created within the visual user interface designer, the object graph comprising the one or more user interface objects.

5. (Previously Presented) A method as recited in claim 1, wherein the serialization manager maintains context information that can be shared among the one or more standard serialization providers and any custom serialization providers that are loaded by the serialization manager.

6. (Original) A method as recited in claim 1, wherein the initial representation comprises a representation used to persist the one or more user interface objects, and wherein the one or more subsequent representations comprise a representation used to represent the one or more user interface objects within the visual user interface designer.

7. (Original) A method as recited in claim 1, wherein the object graph comprises a plurality of related objects.

8. (Cancelled).

9. (Original) A method as recited in claim 1, wherein the one or more standard serialization providers are capable of identifying a plurality of serializers, and wherein at least two of the plurality of serializers are for serializing the object graph in different serialization formats.

10. (Original) A method as recited in claim 1, wherein the serializer produces an eXtensible Markup Language representation of the object graph.

11. (Currently Amended) For a computer system that supports a visual user interface designer and a serialization engine capable of generating source code for user interface objects created within the visual user interface designer, a computer program product comprising one or more computer readable storage media having stored thereon computer executable instructions that implement a method of serializing one or more user interface objects from an initial representation to any of one or more subsequent representations, for one or more standard object types and serialization formats, and which may be extended to cover one or more custom object types and serialization formats, the method comprising acts of:

providing a serialization manager to (i) coordinate one or more standard serialization providers that each identify one or more standard serializers for a standard object type or serialization format, and (ii) load, as needed, one or more custom serialization providers that each identify one or more custom serializers for one or more custom object types or serialization formats;

requesting a serializer from the serialization manager for an object graph that comprises an object of a particular object type, and for a particular serialization format, wherein the serializer is requested from the serialization manager as part of a cut, copy or paste operation and such that due to the cut, copy or paste operation, the serializer produces a snippet of code sufficient to undo or redo a change to the object graph made within the visual user interface designer, but without producing a class representation of the object graph; and

calling the serializer to serialize the object graph.

12. (Original) A computer program product as recited in claim 11, wherein the one or more custom object types or serialization formats are not covered by the one or more standard serialization providers.

13. (Original) A computer program product as recited in claim 11, wherein the serializer is a custom serializer.

14. (Original) A computer program product as recited in claim 13, wherein the serialization manager loads the custom serializer in response to the request.

15. (Previously Presented) A computer program product as recited in claim 11, wherein the serialization manager maintains context information that can be shared among the one or more standard serialization providers and any custom serialization providers that are loaded by the serialization manager.

16. (Original) A computer program product as recited in claim 11, wherein the initial representation comprises a live representation used to represent the one or more user interface objects within the visual user interface designer, and wherein the one or more subsequent representations comprise a target representation used to persist the one or more user interface objects.

17. (Cancelled).

18. (Original) A computer program product as recited in claim 11, wherein the one or more custom serialization providers are capable of identifying a plurality of serializers, and wherein at least two of the plurality of serializers are for serializing the object graph in different serialization formats.

19. (Original) A computer program product as recited in claim 11, wherein the serializer produces a source code representation of the object graph.

20. (Currently Amended) In a computer system that supports a visual user interface designer and a serialization engine capable of generating source code for user interface objects created within the visual user interface designer, a method of serializing one or more user interface objects from an initial representation to any of one or more extensible subsequent representations which may be extended to cover one or more custom object types and serialization formats, the method comprising steps for:

coordinating one or more standard serialization providers that each identify one or more standard serializers for a standard object type or serialization format;

loading, as needed, one or more custom serialization providers that each identify one or more custom serializers for one or more custom object types or serialization formats that are not covered by the one or more standard serialization providers;

identifying a serializer for a particular serialization format and for an object graph that comprises an object of a particular object type; and

serializing the object graph with the identified serializer, wherein serializing the object graph with the identified serializer is performed as part of a cut, copy or paste operation, such that due to the cut, copy or paste operation, the serializer produces a snippet of code sufficient to undo or redo a change to the object graph made within the visual user interface designer, but without producing a class representation of the object graph.

21. (Original) A method as recited in claim 20, wherein the particular object type comprises a custom object type.

22. (Original) A method as recited in claim 20, wherein the particular serialization format is a custom serialization format.

23. (Original) A method as recited in claim 20, further comprising a step for maintaining context information to be shared among the one or more standard serialization providers and any custom serialization providers that are loaded by the serialization manager.

24. (Original) A method as recited in claim 20, wherein the initial representation comprises a live representation used to represent the one or more user interface objects within the visual user interface designer, and wherein the one or more subsequent representations comprise a target representation used to persist the one or more user interface objects.

25. (Original) A method as recited in claim 20, further comprising a step for replacing a standard serializer with a custom serializer from a custom serialization provider loaded while identifying the serializer for the object graph.

26. (Original) A method as recited in claim 20, wherein the object graph comprises a plurality of related objects.

27. (Cancelled).

28. (Original) A method as recited in claim 20, wherein the serializer produces either a source code representation or an eXtensible Markup Language representation of the object graph.

29. (Currently Amended) For a computer system that supports a visual user interface designer and a serialization engine capable of generating source code for user interface objects created within the visual user interface designer, a computer program product comprising one or more computer readable media storage having stored thereon computer executable instructions that implement a method of serializing one or more user interface objects from an initial representation to any of one or more extensible subsequent representations which may be extended to cover one or more custom object types and serialization formats, the method comprising steps for:

coordinating one or more standard serialization providers that each identify one or more standard serializers for a standard object type or serialization format;

loading, as needed, one or more custom serialization providers that each identify one or more custom serializers for one or more custom object types or serialization formats that are not covered by the one or more standard serialization providers;

identifying a serializer for a particular serialization format and for an object graph that comprises an object of a particular object type; and

serializing the object graph with the identified serializer, wherein serializing the object graph with the identified serializer is performed as part of a cut, copy or paste operation, such that due to the cut, copy or paste operation, the serializer produces a snippet of code sufficient to undo or redo a change to the object graph made within the visual user interface designer, but without producing a class representation of the object graph.

30. (Original) A computer program product as recited in claim 29, wherein one or more custom serialization providers are loaded in identifying the serializer for the object graph.

31. (Original) A computer program product as recited in claim 29, the method further comprising a step for maintaining context information to be shared among the one or more standard serialization providers and any custom serialization providers that are loaded by the serialization manager.

32. (Original) A computer program product as recited in claim 29, wherein the initial representation comprises a representation used to persist the one or more user interface objects, and wherein the one or more subsequent representations comprise a representation used to represent the one or more user interface objects within the visual user interface designer.

33. (Original) A computer program product as recited in claim 29, the method further comprising a step for replacing a standard serializer with a custom serializer from a custom serialization provider loaded while identifying the serializer for the object graph.

34. (Cancelled).

35. (Original) A computer program product as recited in claim 29, wherein the serializer produces either a source code representation or an eXtensible Markup Language representation of the object graph.

36. (Previously Presented) For a computer system that supports a visual user interface designer and a serialization engine capable of generating source code for user interface objects created within the visual user interface designer, a computer program product comprising one or more computer readable storage media having stored thereon computer executable instructions that implement a method of efficiently serializing one or more user interface objects when performing visual operations on the one or more user interface objects, the method comprising steps for:

loading one or more serialization providers that each identify one or more serializers for a given object type and serialization format;

from the one or more serialization providers, identifying a serializer capable of serializing a user interface object in a serialization format that corresponds to a requested visual operation to be performed on the user interface object; and

serializing the user interface object with the identified serializer, wherein serializing the user interface object with the identified serializer is performed as part of a cut, copy, paste, undo or redo operation, such that due to the cut, copy, paste, undo or redo operation, the serializer produces a snippet of code sufficient to undo or redo a change to the user interface object made within the visual user interface designer without producing a class representation of the user interface object in order to efficiently perform the requested operation.

37. (Original) A computer program product as recited in claim 36, further comprising a step for coordinating the one or more serialization providers.

38. (Original) A computer program product as recited in claim 36, wherein the one or more serialization providers comprise one or more standard serialization providers that each identify one or more standard serializers for a standard object type and serialization format.

39. (Original) A computer program product as recited in claim 38, wherein the one or more serialization providers comprise one or more custom serialization providers that each identify one or more custom serializers for one or more custom object types or serialization formats that are not covered by the one or more standard serialization providers.

40. (Original) A computer program product as recited in claim 39, wherein one or more custom serialization providers are loaded in identifying the serializer.

41. (Original) A computer program product as recited in claim 36, wherein the serializer is a custom serializer.

42. (Original) A computer program product as recited in claim 36, the method further comprising a step for maintaining context information to be shared among the one or more serialization providers.

43. (Cancelled).

44. (Cancelled).

45. (Original) A computer program product as recited in claim 36, the method further comprising steps for:

from the one or more serialization providers, identifying a code serializer capable of serializing a user interface object in a serialization format that corresponds to a source code representation of the user interface object; and

serializing the user interface object with the identified code serializer to produce a class representation of the user interface object.

46. (Original) A computer program product as recited in claim 36, the method further comprising steps for:

from the one or more serialization providers, identifying an eXtensible Markup Language (XML) serializer capable of serializing a user interface object in a serialization format that corresponds to an XML representation of the user interface object; and

serializing the user interface object with the identified XML serializer to produce an XML representation of the user interface object.